BAB 05

COOKIE

Aplikasi web adalah aplikasi yang stateless, di mana server tidak akan mengingat request client. Akibatnya, setiap request yang masuk selalu dianggap oleh server berasal dari client-client yang berbeda. Padahal, kadang-kadang server juga perlu mengenali client, misal untuk kepentingan authentikasi di mana client tidak bisa mengakses sembarang halaman yang tidak diizinkan. Untuk mengatasi masalah ini, bisa digunakan cookie dan session.

Pada bab ini kita khusus membahas mengenai cookie, sedang untuk session akan kita bahas pada bab selanjutnya.

5.1 Cookie

Cookie merupakan informasi kecil yang dikirim kepada client/ browser dari server. Cookie dijadikan tempat penyimpanan sementara state yang disimpan di sisi client/browser. Oleh browser, cookie akan disimpan dan nantinya dapat dikirimkan kembali ke server jika diperlukan.

Biasanya cookie dipakai untuk mengidentifikasi client. Cookie berisi suatu ID yang unik pada tiap client. Bagaimana server bisa mengenali banyak client yang pada saat bersamaan sedang melakukan request.

Dalam kasus tertentu misalnya website yang mewajibkan user untuk login supaya dapat mengakses resource-resource dari website tersebut, tentunya akan repot jika user harus terus-menerus

melakukan login setiap mengakses halaman website yang berbeda. Oleh karena itu, server dapat mengirimkan cookie berupa identitas masing-masing client, sehingga dapat dikenali apakah client yang bersangkutan sudah melakukan login atau belum.

Cookie disimpan di browser. Browser yang berbeda dianggap sebagai client yang berbeda dan cookie bisa dihapus melalui browser masing-masing. Mungkin kita pernah mengakses sebuah website, di mana kita atur menggunakan bahasa Indonesia dalam website itu, dan ketika kita mengakses kembali website dengan browser yang sama maka secara otomatis website tersebut langsung menampilkan halaman berbahasa Indonesia. Ini terjadi karena browser menyimpan state penggunaan bahasa kita. Namun, ketika kita mengakses dengan browser yang lain, kita harus mengatur kembali penggunaan bahasanya karena kita sudah dianggap client yang berbeda.

Cookie hanya dapat menampung data string sederhana bukan data kompleks seperti context. Cookie akan terhapus begitu browser ditutup, namun cookie dapat diatur waktu hidupnya sehingga bisa disimpan lebih lama oleh web browser. Selain itu, cookie hanya diubah oleh server dari mana cookie tersebut berasal.

5.2 Keuntungan dan Kelemahan Cookie

Dengan karekteristik cookie seperti yang sudah dijelaskan sebelumnya, maka cookie memiliki beberapa keuntungan sekaligus kelemahan.

Kelemahan cookie yang utama, yaitu karena cookie disimpan oleh client, maka setiap client yang ingin mengakses website yang membutuhkan cookie, wajib menggunakan browser yang mendukung cookie dan mengaktifkan cookie-nya. Jika tidak, aplikasi web tidak akan berjalan sebagaimana mestinya. Ini adalah kelemahan di mana aplikasi web menjadi sangat bergantung dengan client itu sendiri.

Meski hanya bisa dimodifikasi oleh server, namun cookie bisa dibaca oleh unauthorized user. Cookie dapat meningkatkan beban di jaringan akibat informasi yang sama yang dikirim terus-menerus. Namun, beban ini biasanya tidak akan terasa karena ukuran cookie sendiri yang sangat kecil, yaitu sekitar 4kb.

Keuntungan penggunaan cookie selain bisa digunakan untuk identifikasi client adalah cookie tidak terpengaruh pada crash server karena data disimpan di client serta dapat meringankan beban di memori user.

5.3 Cookie di Java

Di java sudah disediakan fasilitas untuk menggunakan cookie melalui class javax.servlet.http.Cookie.

Untuk menciptakan cookie, berikut langkah-langkahnya:

- Panggil constructor Cookie. Misal seperti ini:

  Cookie c = new Cookie("ckieNama", "luwis");

  String "ckieNama" adalah nama cookie-nya. Sedangkan string "luwis" adalah isi cookie-nya. Nama dan isi cookie tidak boleh mengandung spasi, tab, dan karakter seperti: { [] ( ) = , ” / @ : ;

- Agar cookie tidak terhapus begitu browser ditutup dan masih bisa digunakan lagi, atur umur maksimum cookie-nya.

  *c.setMaxAge(60*60*24);*

  Berarti umur maksimum cookie kita adalah 60 x 60 x 24 detik atau 1 hari. Setelah itu cookie akan terhapus.

- Untuk mengirimkan cookie kepada client, tambahkan cookie kepada object response.

  response.addCookie(c);

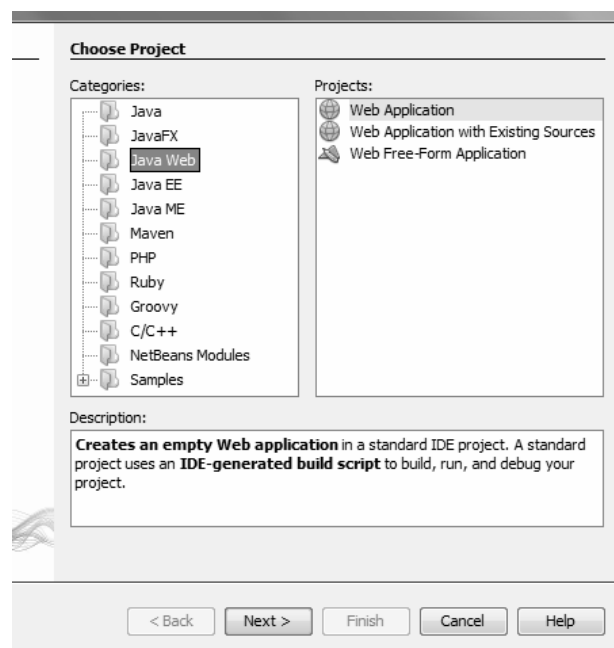
Untuk membaca cookie gunakan method *request.getCookie()*. Nilai kembalian dari method ini adalah array object cookie yang disimpan oleh browser dan akan mengembalikan null jika tidak ada cookie.

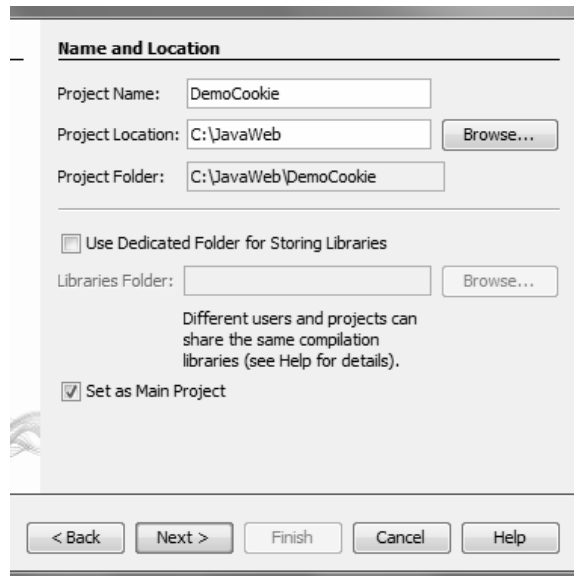
Untuk mendapatkan cookie yang kita maksud, lakukan perulangan menggunakan method *getName()* pada setiap object cookie sampai ditemukan cookie yang dimaksud.

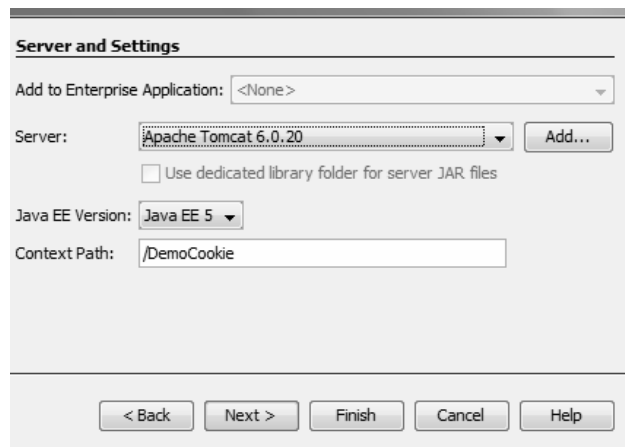
5.4 Contoh Program

Setelah kita mengetahui apa itu cookie, kegunaan, kelemahan dan cara menggunakan cookie, sekarang kita coba cookie dalam program.

Skenarionya kita buat sebuah halaman login dan sebuah halaman khusus untuk member. Jika kita login sebagai member, kita akan diizinkan mengakses halaman khusus member tersebut, dan jika bukan member maka kita tidak diizinkan. User member maupun non-member akan memperoleh id masing-masing. Sehingga jika ada user non-member yang ingin langsung mengakses halaman khusus member tanpa login maka akan ditolak.

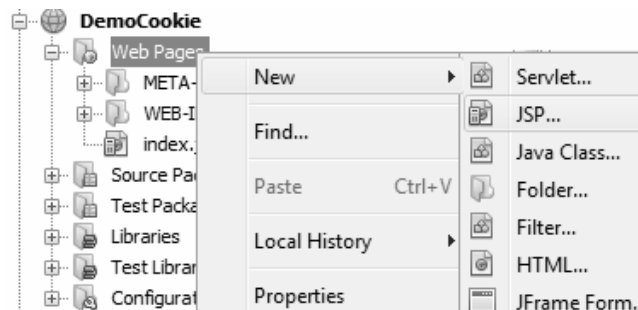
1. Kita buat sebuah project baru. Klik menu File > New Project.
2. Pilih category: Java Web dan projects: Web Application.

Gambar 5.1 Memilih project Web Application

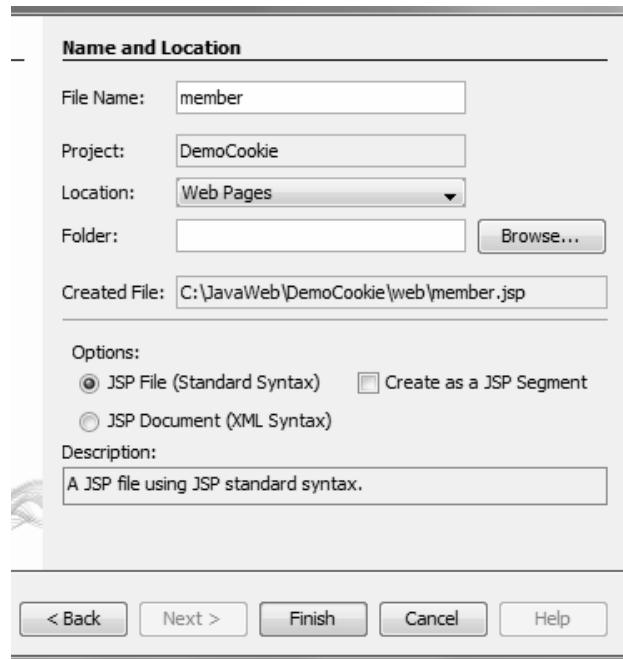
3. Beri nama project: DemoCookie. Klik Next.

Gambar 5.2 Membuat project DemoCookie

4. Pastikan server-nya adalah Apache Tomcat dan java ee version-nya adalah Java EE 5. Kemudian langsung klik Finish.

Gambar 5.3 Memilih server Apache Tomcat

5. Buat sebuah form input username dan password pada index.jsp. Ketik code berikut.

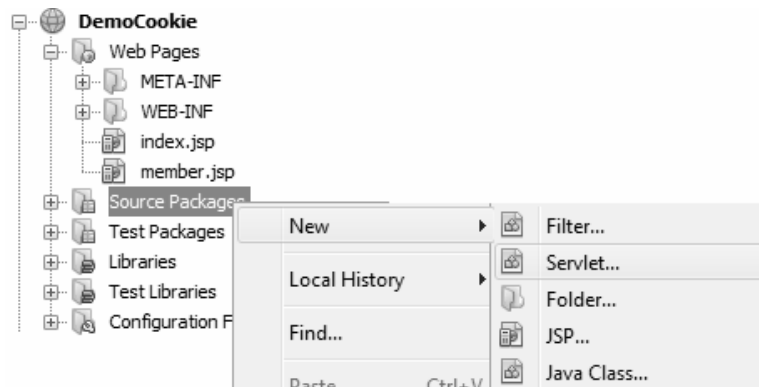
```
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html;
    charset=UTF-8">
    <title>JSP Page</title>
    </head>
<body>
    <h2>Member Login</h2>
    <form method='post' action='login'>
            <fieldset>
                    Username:
                    <input type='text' name='uname' />
                    <br><br>
                    Password:
                    <input type='password' name='pass' />
                    <br><br>
                    <input type='submit' value='Login' />
            </fieldset>
    </form>
</body>
</html>
```

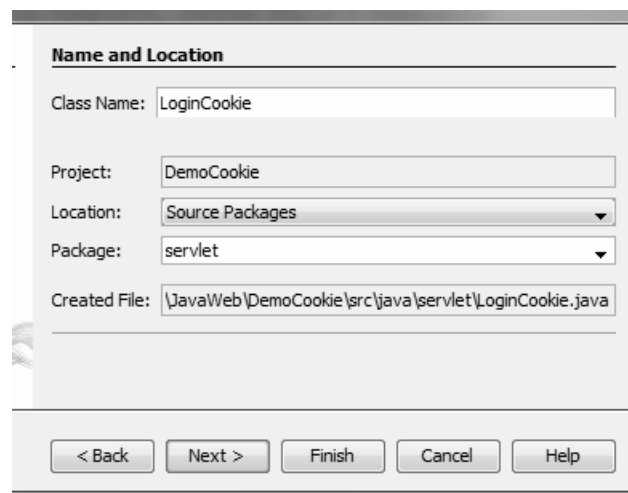
6. Kita buat lagi sebuah halaman jsp. Klik kanan pada Web Pages di tab project. Pilih New > JSP.

Gambar 5.4 Membuat file jsp baru

7. Beri nama file: member. Klik Finish.

Gambar 5.5 Membuat member.jsp

8. Ketik code berikut pada member.jsp.

```
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html;
    charset=UTF-8">
    <title>JSP Page</title>
</head>
<body>
    <h1>Halaman Khusus Member</h1>
</body>
</html>
```

9. Kita buat sebuah servlet baru untuk menambah cookie. Klik kanan pada source packages di tab project. Pilih New > Servlet.

Gambar 5.6 Membuat servlet baru

10. Beri nama servlet: LoginCookie. Isikan package: servlet. Klik Next.

Gambar 5.7 Membuat servlet LoginCookie

11. Ubah url pattern-nya menjadi /login. Klik Finish.

Gambar 5.8 Configure URL pattern servlet LoginCookie

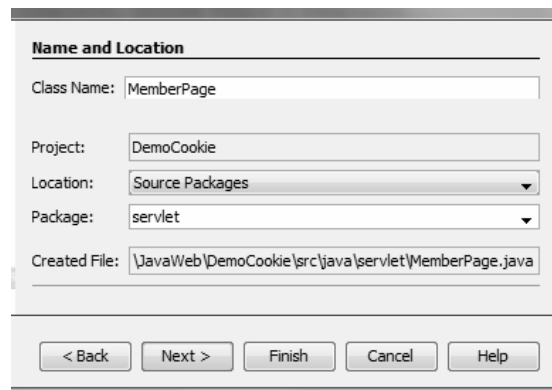
12. Ketik code berikut dalam method processRequest().

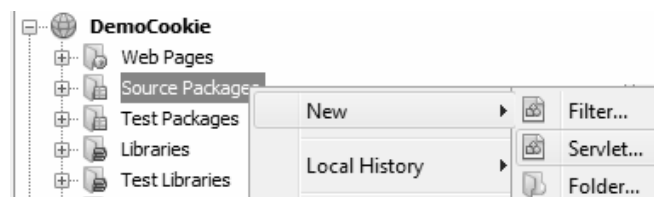
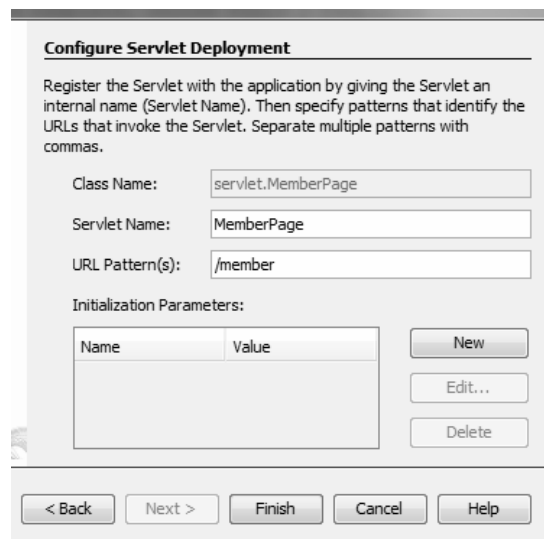
```
protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();
    String username = request.getParameter("uname");
    String password = request.getParameter("pass");
    String cookieMember = "1234567890";
    String cookieNonMember = "000";
    if(username.equalsIgnoreCase("member") &&
    password.equals("member")){
            Cookie c = new Cookie("ckieNama", cookieMember);
            c.setMaxAge(60*60);
            response.addCookie(c);
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Login</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h2>Login benar!!!</h2>");
            out.println("<h3><a href=\"member\">Klik untuk
            menuju ke halaman member...</a></h3>");
            out.println("</body>");
            out.println("</html>");
```

```
    }else{
            Cookie c = new Cookie("ckieNama",
            cookieNonMember);
            c.setMaxAge(60*60);
            response.addCookie(c);
            out.println("Login Salah!!");
            RequestDispatcher dis = null;
            dis = request.getRequestDispatcher("/index.jsp");
            dis.include(request, response);
    }
}
```

Program ambil nilai dari form input username dan password. Username dan password ini akan divalidasi.

- Jika username dan password-nya adalah "member" maka cookie ckieNama akan bernilai konstanta cookieMember dan akan menampilkan halaman "Login benar" serta link untuk memasuki halaman member.
- Jika username dan password bukan "member" maka cookie ckieNama akan bernilai konstanta cookieNonMember dan akan dikembalikan ke index.jsp dengan peringatan "Login salah".

Cookie diset berumur 60x60 detik atau 1 jam. Setelah durasi request mencapai 1 jam maka cookie akan terhapus.

Di dalam aplikasi profesional, isi cookie jangan berupa konstanta. Isi cookie haruslah acak dan unik untuk client tertentu, bisa saja berdasarkan username dan password.

13. Kita buat lagi sebuah servlet untuk mengarahkan user ke halaman member. Klik kanan pada source packages di tab project. Pilih New > Servlet.

Gambar 5.9 Membuat servlet baru

14. Beri nama servlet: MemberPage. Isikan package: servlet. Klik Next.

Gambar 5.10 Membuat servlet MemberPage

15. Ubah url pattern-nya menjadi /member. Klik Finish.

Gambar 5.11 Configure URL pattern servlet MemberPage

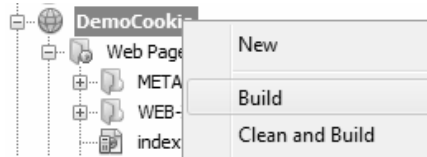
16. Ketik code berikut dalam method processRequest().

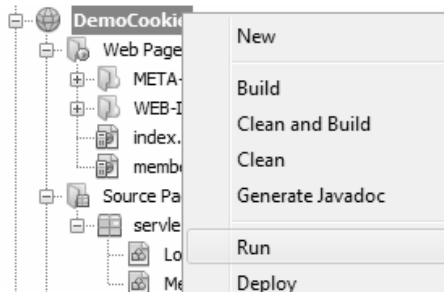
```
protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();
    RequestDispatcher dis;
    String cookieMember = "1234567890";
    try {
        Cookie[] cookies = request.getCookies();
        String cNama = "";
        if(cookies!=null){
            for(int x=0; x<cookies.length; x++){
                if(cookies[x].getName().equals("ckieNama")){
                    cNama = cookies[x].getValue();
                    break;
                }
            }
        }
        if(cNama.equals(cookieMember)){
            dis =
            request.getRequestDispatcher("/member.jsp");
        } else {
            dis = request.getRequestDispatcher("/index.jsp");
            out.println("Anda Belum Login Sebagai Member!!");
        }
        dis.include(request, response);
    } finally {
        out.close();
    }
}
```

Di servlet MemberPage ini program akan mengecek lagi validitas dari client. Mengapa kita cek lagi validitas client-nya? Untuk mencegah client non-member yang ingin masuk ke halaman member tanpa mekanisme login terlebih dahulu.

Program akan mengecek semua cookie pada client. Jika ada cookie dengan nama ckieNama, selanjutnya akan diperiksa isi cookie tersebut.

- Jika isi cookie ckieNama adalah konstanta cookieMember maka client akan diarahkan ke halaman khusus member (member.jsp).
- Jika isi cookie ckieNama bukan konstanta cookieMember maka client akan diarahkan ke halaman index.jsp dengan peringatan "Anda belum login sebagai member".

17. Jalankan project DemoCookie. Klik kanan project DemoServletContext > Build.

Gambar 5.12 Build project DemoCookie

18. Tunggu hingga proses build selesai. Klik kanan project DemoCookie > Run.

Gambar 5.13 Run project DemoCookie

19. Masuklah sebagai member. Ketik username dan password "member".

Gambar 5.14 Login sebagai member

20. Jika login benar akan muncul tampilan berikut.

Login benar!!!

Klik untuk menuju ke halaman member...

Gambar 5.15 Berhasil masuk sebagai member

21. Klik link-nya kita akan masuk ke halaman member.

Halaman Khusus Member

Gambar 5.16 Halaman khusus member

22. Kita sudah masuk sebagai member. Kita coba untuk langsung mengakses halaman member tanpa melalui mekanisme login. Buka tab baru dan ketikkan url berikut.

    http://localhost:8084/DemoCookie/member

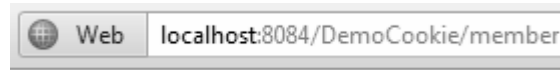
Gambar 5.17 URL halaman member

23. Hasilnya kita akan tetap bisa mengakses halaman khusus member.

Halaman Khusus Member

Gambar 5.18 Halaman khusus member

24. Tapi, cobalah membuka browser lain dan masuk langsung ke halaman khusus member. Ketik url berikut.

    http://localhost:8084/DemoCookie/member

Gambar 5.19 URL halaman member

25. Kita tidak bisa mengakses halaman member karena dianggap belum login disebabkan browser berbeda, berarti client berbeda pula.

Gambar 5.20 Halaman index dengan peringatan user belum login sebagai member

26. Sekarang kita coba untuk masuk bukan sebagai member. Ketikkan username dan password sembarang.

Gambar 5.21 Login bukan sebagai member

27. Hasilnya akan dikembalikan ke index dengan peringatan login salah.

Gambar 5.22 Peringatan login salah

28. Kita belum masuk sebagai member. Kita coba lagi untuk langsung mengakses halaman member tanpa melalui mekanisme login. Buka tab baru dan ketikkan url berikut.

    http://localhost:8084/DemoCookie/member

Gambar 5.23 URL halaman member

29. Hasilnya akan dikembalikan ke index dengan peringatan bahwa kita belum masuk sebagai member. Ini terjadi karena isi cookie kita sudah berubah, bukan cookie member lagi.

Gambar 5.24 Halaman index dengan peringatan user belum login sebagai member